horizontal line

## Balaji Betadur

Email: balajibetadur@gmail.com

Phone: 8892011406

[Portfolio](http://balajibetadur.github.io/)

AI **EMAIL CLASSIFIER**

## Problem Definition

A multinational Company is trying to streamline its customer care process. For this, they have to categorize customer query mails. Participants have to build a re-trainable AI model to classify emails based on bulk emails as a training dataset.

**TABLE OF CONTENTS**

1. [Introduction](#_ifda1x9yfooh)
2. [Proposed System](#_thag407a49v6)
   1. Training Data creation model
   2. Classifier module
      1. Re-training classifier module
      2. Predicting / Classifying test data
   3. Run time module
      1. Training Model
      2. Testing new data
3. [Technology Stack](#_2z1rtbh6bstv)
   1. Technologies
   2. Packages
   3. Tools
4. [Use case Diagram](#_tzn7nilmlbwy)

## Introduction

In today’s generation, most of the professional conversations are through emails. But it also lets people reach anyone they want with their email id. This might lead to a problem that the important mails may get lost. The concept of spam detection helped to solve this issue, where all the spam messages get filtered out.

Similarly, we can also classify emails into different categories to give more importance to the priority email of that specific category. It helps to categorize all the emails based on different features. One of the techniques to categorize the emails is Text Classification.

|  |
| --- |
| **What is text classification?**  Text classification also known as text tagging or text categorization is the process of categorizing text into organized groups. By using Natural Language Processing (NLP), text classifiers can automatically analyze text and then assign a set of pre-defined tags or categories based on its content. [know more](https://medium.com/data-from-the-trenches/text-classification-the-first-step-toward-nlp-mastery-f5f95d525d73) |

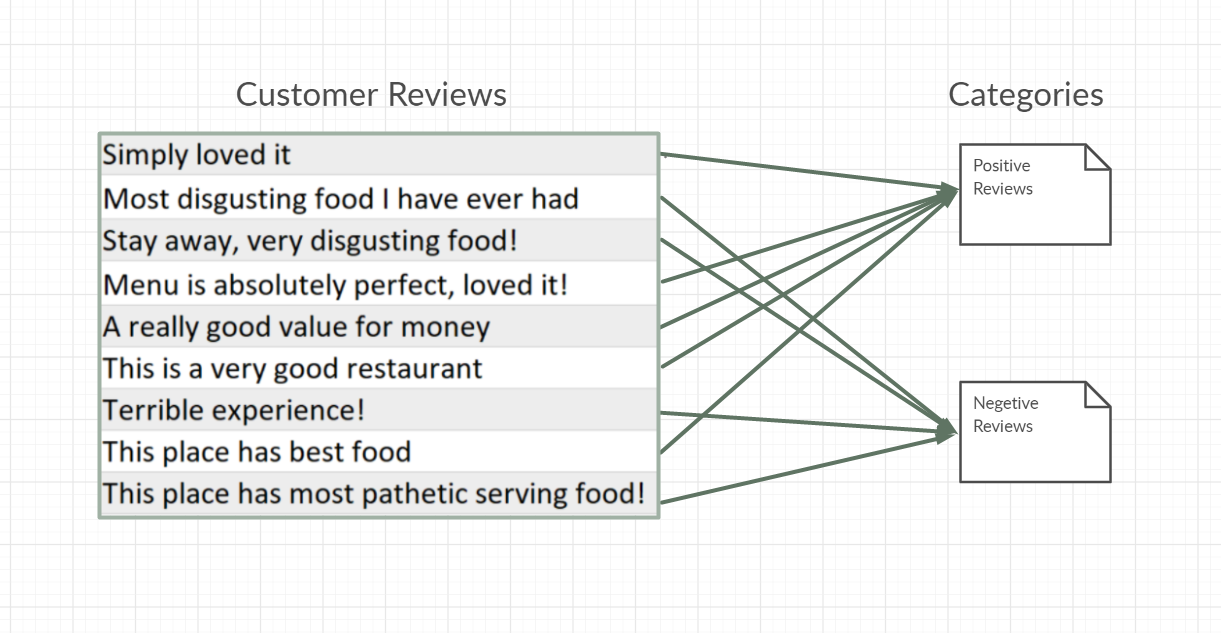
Fig 1 shows the simple text classification task using customer reviews as negative and positive reviews, emails can also be classified similarly.

Fig 1: Illustration of Text classification for a Hotel using Customer Reviews

* 1. **Directory Structure**
  2. Documentation Folder : Includes documentation, report, workflow, flowchart, algorithm diagrams
  3. Models Folder : Includes all saved trained models
  4. Static Folder: Includes js files
  5. Templates Folder: Includes html files i.e. index.html
  6. Tests Folder: Includes data uploaded for prediction
  7. Upload Folder : Includes data uploaded for training.
  8. App.py file : It is the application file where trainng and prediction code is included.
  9. Models.py file : This file includes all algorithms.
  10. Preprocess.py file : It includes data processing functions.
  11. Data\_preprocessed.xlsx file : It is the output of preprocess.py i.e processed data
  12. Results.csv : This the downloaded result file after prediction of test data

## Proposed system

Today, **AI** can perform intensive human labor and backbreaking tasks easily without the need for human intervention. We are using AI (Machine Learning and Deep Learning) to find a solution for categorizing emails.

There are three modules in this solution:

1. Training data creation module
2. Classifier module
3. Run-time component

**2.1 Training data creation module**

The user uploads data in .eml format which needs to be preprocessed. The raw data can be converted to text or CSV using python. Package ‘email’ helps to fetch the data from the eml file.‘ Email can also have attachments like pdf’s, excel files, or any other documents, which will add value to the model to categorize the email.

|  |
| --- |
| **What is EML?**  **EML** is a **file extension** for an e-mail message saved to a **file** in the MIME RFC 822 standard **format** by Microsoft Outlook Express as well as some other email programs. **EML files** can contain plain ASCII text for the headers and the main message body as well as hyperlinks and attachments. [know more](https://whatis.techtarget.com/fileformat/EML-Microsoft-Outlook-Express-mail-message-MIME-RFC-822#:~:text=EML%20is%20a%20file%20extension,well%20as%20hyperlinks%20and%20attachments.) |

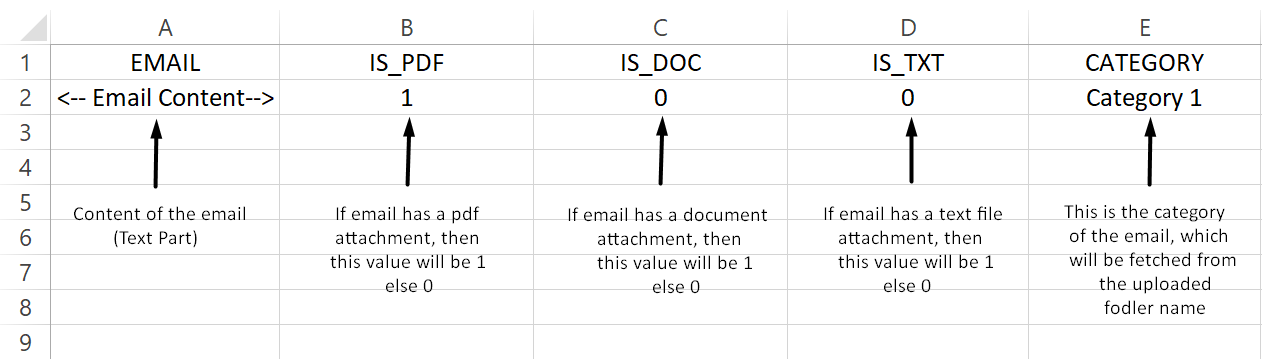


Fig 2: Converted sample data in .csv format

**2.2 Classifier module**

Machine Learning and Deep Learning have different algorithms for the classification of text. The user can train the model of his choice based on criteria like training time, accuracy, testing time, etc.

**2.2.1 Re-Training Classifier module**

The flexibility of the solution is very important as the data keeps on changing all the time. The user can upload new data and train the model of his choice. The 2 primary options are

1. Training time (low, moderate)
2. Algorithm (based on training time)

If the user selects training time as ‘low’ then only machine learning algorithms are displayed in the algorithms section as they consume less time to train compared to deep learning algorithms else if the selection is ‘moderate’ then both machine learning and deep learning algorithms are displayed. Fig 1 shows the workflow of the algorithm.

Based on the training time, algorithms are listed in a dropdown. Users can select any algorithm among them.

The user can leave other values as default or he can change them in the advanced configuration which includes Number of epochs, split size, learning rate, batch size, etc (parameters vary with an algorithm). Once the user starts training the model, the expected time of training is displayed based on the algorithm, and other parameters selected by the user.

|  |
| --- |
| **What if the user is from a non-technical background?**   * For every parameter, there is a default value based on certain criteria. * Before training, the algorithm is shortlisted based on training data (Data size, Number of classes). * While training, the parameters are updated like Learning rate from learning rate curve, from learning rate decay, the number of epochs from early stopping, etc. * Parameters vary with the algorithm, different parameters are displayed based on the selected algorithm. |

Once Training is completed the user can download the trained model. Incase testing is required at different points of time, the user can upload the model to avoid training again and also to prevent any loss of data.

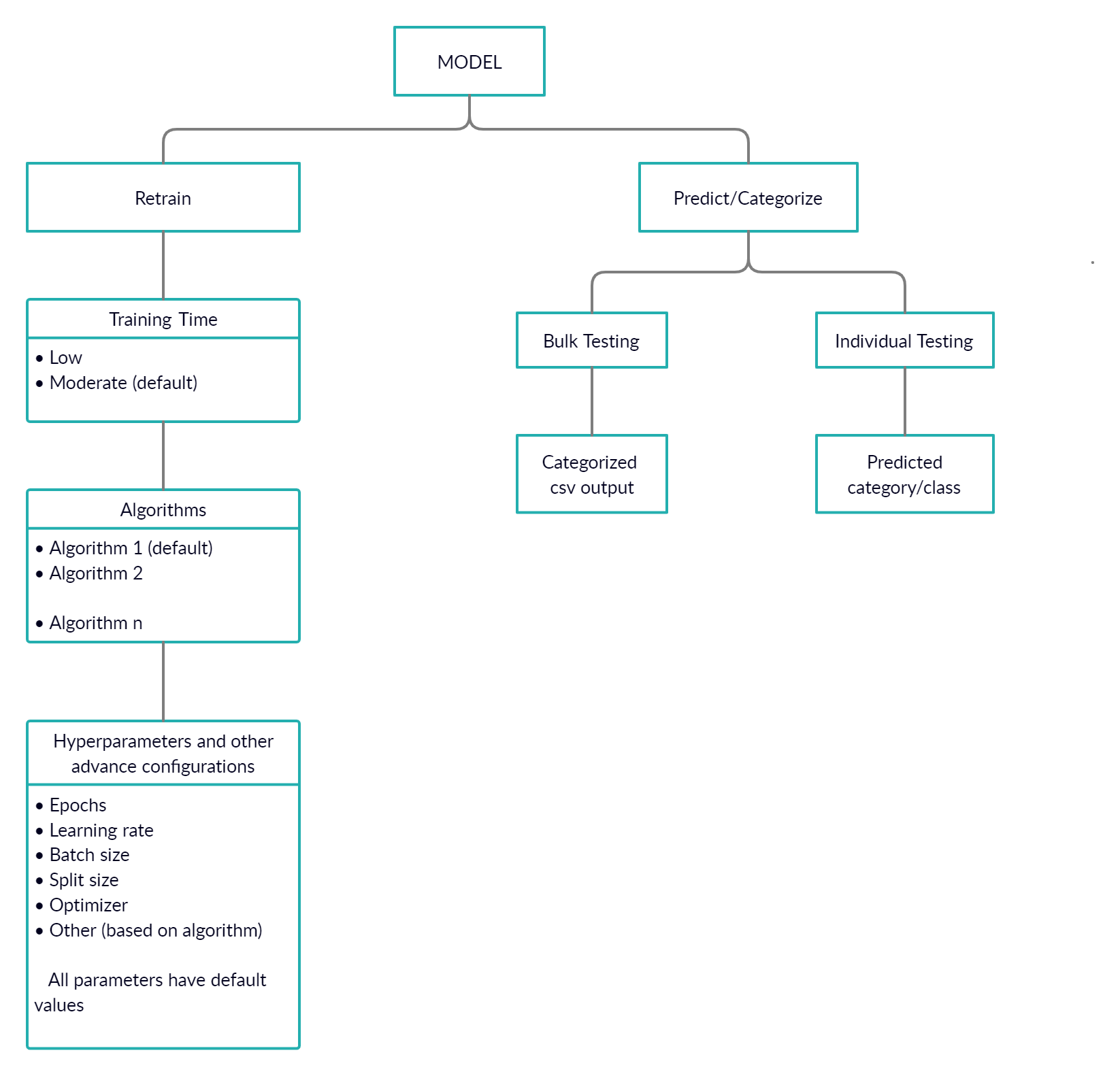


Fig 3: Workflow including both re-training and predicting

Once Training is completed the user can download the trained model. Incase testing is required at different points of time, the user can upload the model to avoid training again and also to prevent any loss of data.

Different algorithms perform well on different datasets. As data will not be the same every time multiple algorithms are considered.

|  |  |
| --- | --- |
| Algorithm | Reason |
| Logistic Regression | Suits small Data, Perform well for binary classification |
| Random Forest | High accuracy for both small and moderate dataset |
| SVM | Works relatively well when there is a clear margin of separation between classes, memory efficiency. |
| XG Boost | Suitable for structured data and gives better performance in classification tasks. |
| AdaBoost | Perform well in binary classification, low generalization error. |
| LSTM | Creates both short-term and long-term memory components, suits for moderate and large datasets gives and gives high accuracy. |
| BERT | Context understanding, High Accuracy, suits for small, moderate, and large datasets |

Table 1: Algorithms list along with the reason why they can be used

**2.2.2 Classifying/Categorizing the test data**

Once Training is completed the user can upload new testing data for categorizing. Options for both bulk email uploading and single email testing are available. In the case of bulk testing, the model generates and displays a CSV file in which samples and their respective categories are listed, and for a single email, the category is displayed instantly. Users can download the CSV after training completion.

**2.3 Runtime Module**

The runtime component is simple to use and has a clean User Interface. It mainly consists of 2 sections, one for training models and another for predicting or classifying test data. The User Interface and the operation of the entire system are recorded and uploaded [here](https://drive.google.com/file/d/1btUoESQlCoM4OAOax9CBRHpgTOCQz7RM/view?usp=sharing)**.**

Fig 4 shows the simple UI where we have 2 sections namely

1. Train Model
2. Predict / Categorize

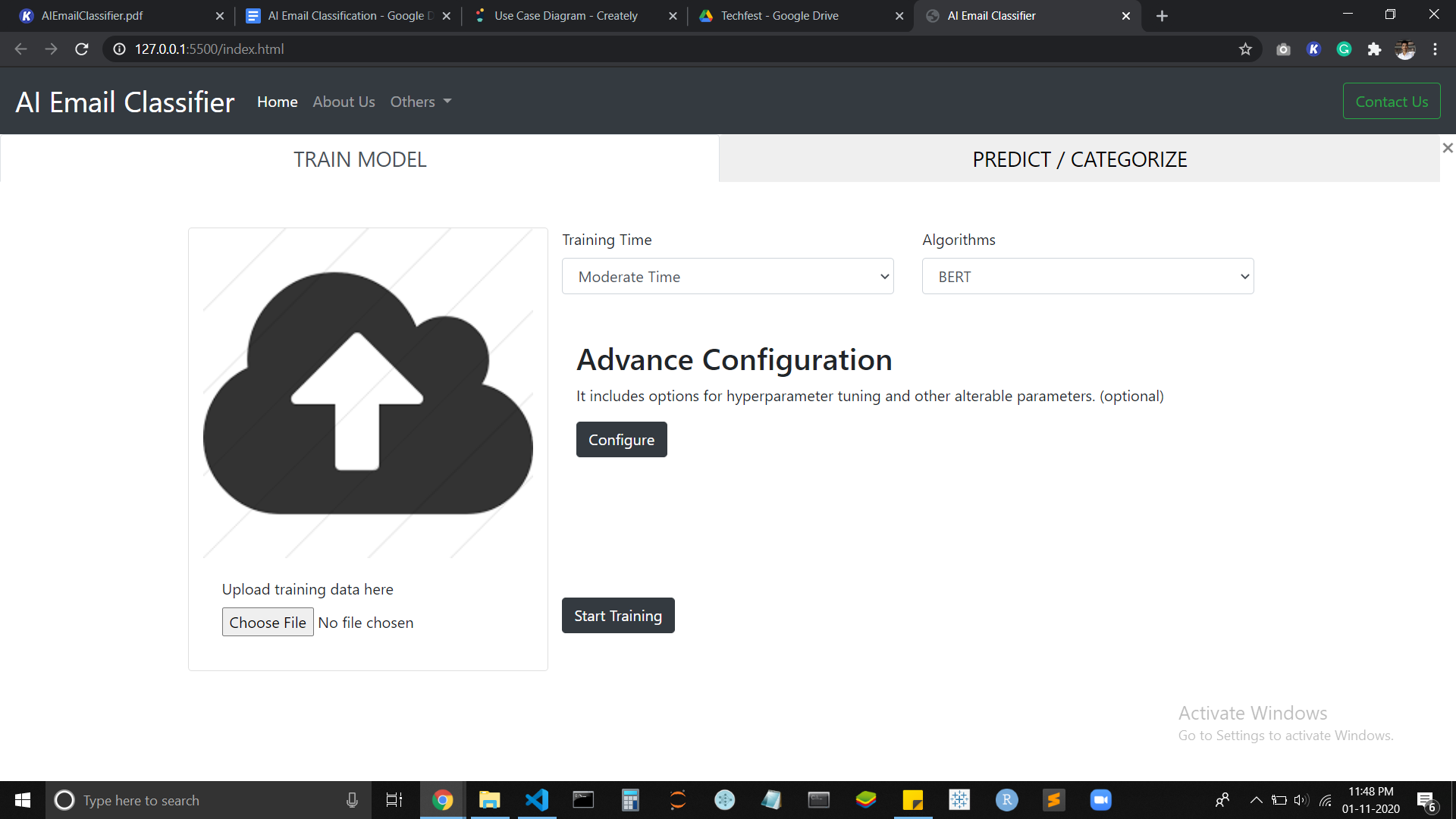


Fig 4: Simple two sectioned UI.

**2.3.1 Training Model**

In the Training section, there are 2 basic options to select, Training time and algorithm (optional). Training time has 2 options low and moderate. Based on the training time the algorithms are displayed.

There is another section called advanced configuration where we have other options for hyperparameter tuning. Fig 5 shows the advanced configuration options. This section includes basic parameters like split size, learning rate, number of epochs, etc. These options vary based on the algorithm selected by the user. Each algorithm has a set of parameters. You can also upload a JSON file with the specified format for hyperparameter tuning.

Once the user clicks start training, the training data(zip file) uploaded data is converted into a required CSV format as shown in Fig 2. All the parameters selected by the user are used to train the model. Every option provided is optional and has a default value in case the user does not choose any value for that option.

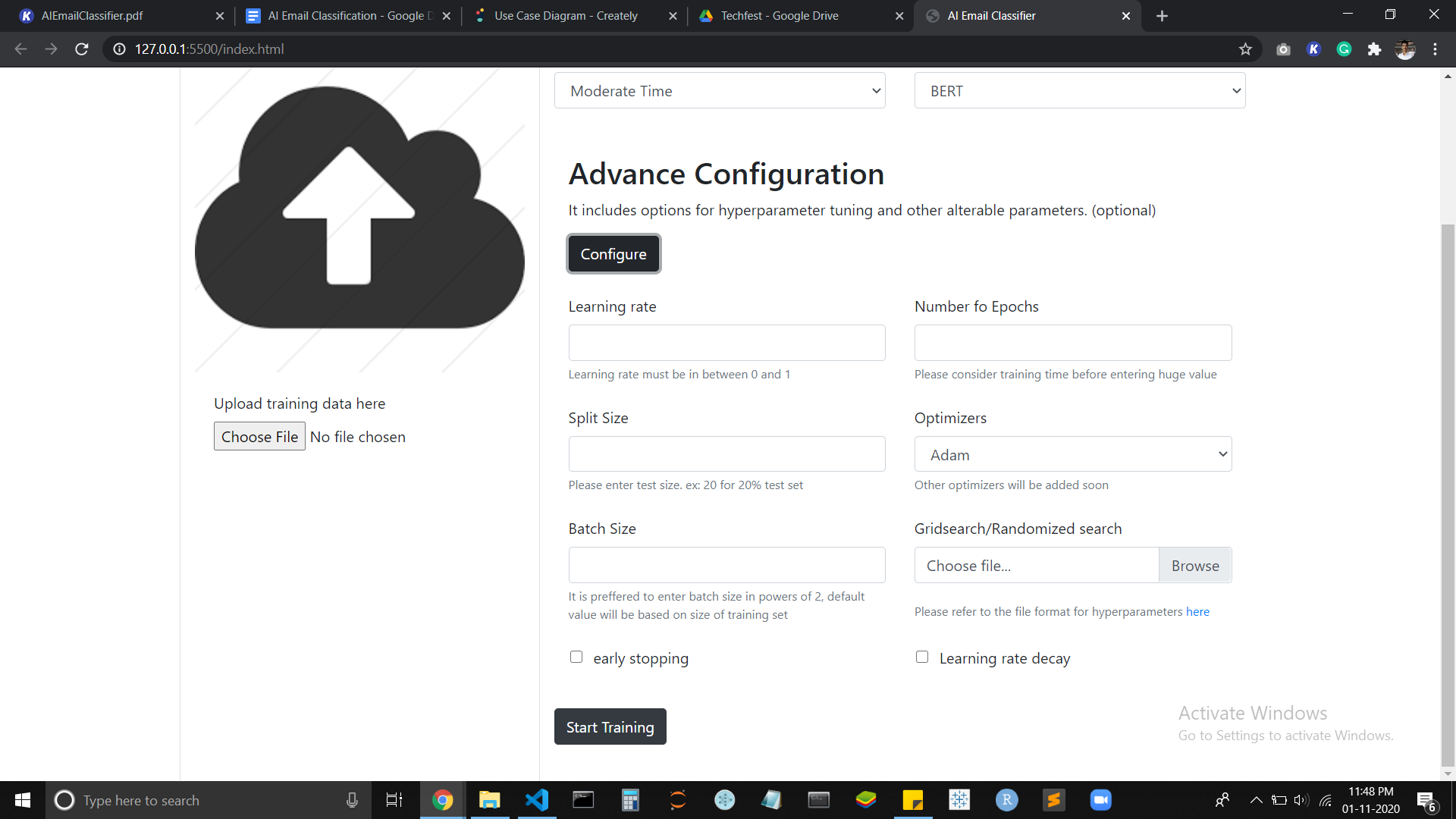


Fig 5: Advance configuration for hyperparameter tuning.

**2.3.2 Predicting test data**

Once the training is complete, new test data can be tested both in bulk or individually. Fig 6 shows the interface for testing new data.

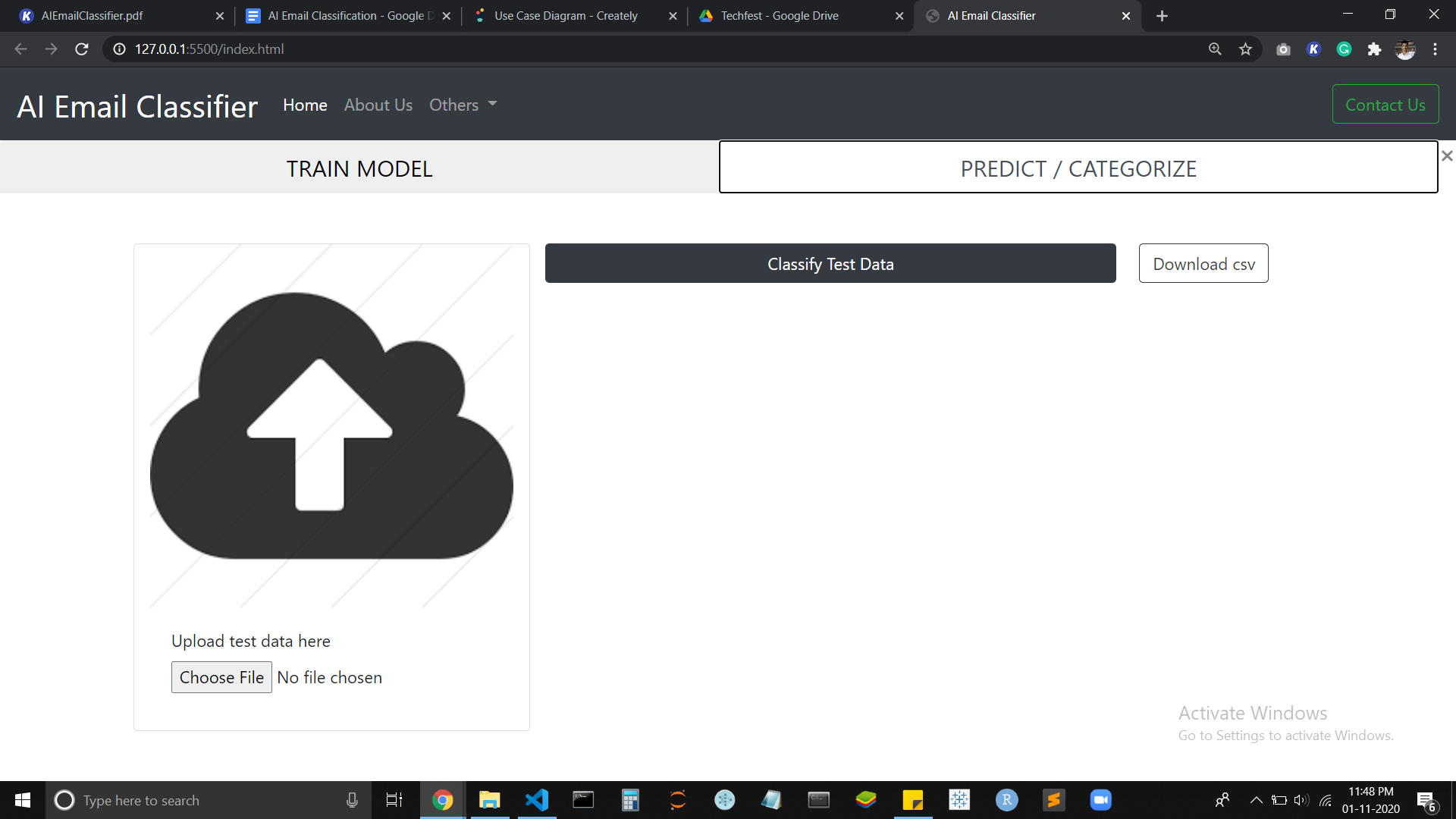


Fig 6: UI for testing module

The user can upload a single file or a bulk of testing files for testing/classifying. After uploading the data, you just need to click classify test data. The result is displayed as shown in Fig 7.

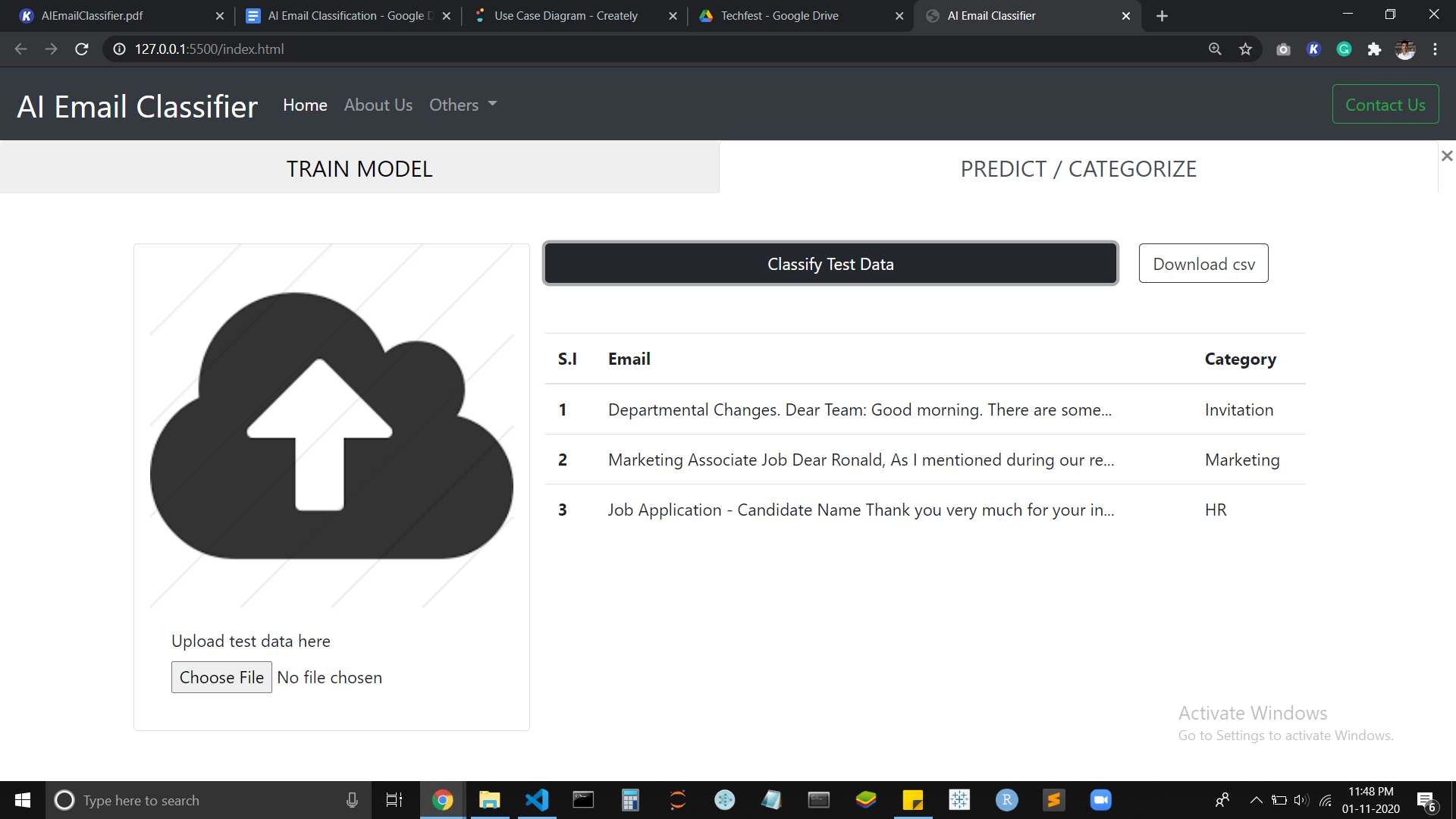


Fig 7: Result of testing data and option to download a generated result CSV file

|  |
| --- |
| **How does a website work? Is it user friendly?**   * The demo of a sample UI for this project is recorded and uploaded [here](https://drive.google.com/file/d/1JYgYldsRN5qM65WGS3uNDW-fuce-Bq5a/view?usp=sharing). |

## Technology Stack

This is an end-to-end project, starting from data preparation up to Deployment we use different technologies. Python is the major programming language for both back-end scripting and model training.

* List of all technologies used:

1. Python : Core programming language
2. Flask : Python micro web-framework for back-end
3. Bootstrap : Front-end templates
4. Javascript : Front-end scripting
5. Git : Version control system
6. Heroku : Cloud platform for deployment

* List of all Python packages used:

1. Keras
2. Fastai
3. Pytorch
4. Flask
5. Pandas
6. JSON
7. Pickle
8. Scikit-learn
9. K-train
10. Os
11. Time
12. Datetime
13. Scipy
14. Numpy
15. Jinja2

* List of all Tools used:

1. Visual Studio Code
2. Jupyter Notebook
3. Google Colab
4. Git
5. Heroku

|  |
| --- |
| **The usage of the tools, packages, and technologies are not restricted to the above mentioned lists.** |

## Use case DIagram

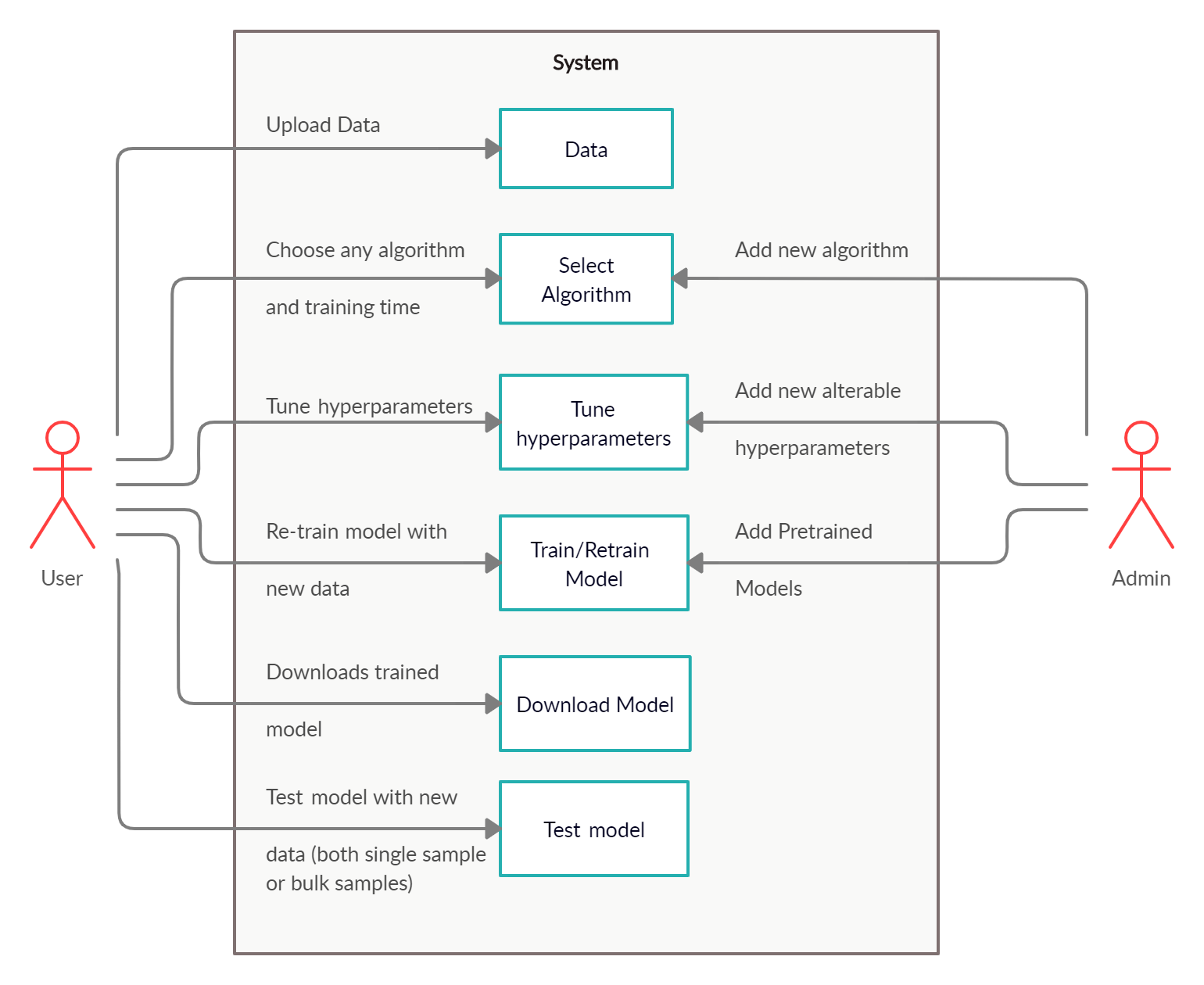


Fig 8: Use Case Diagram of AI Email Classifier

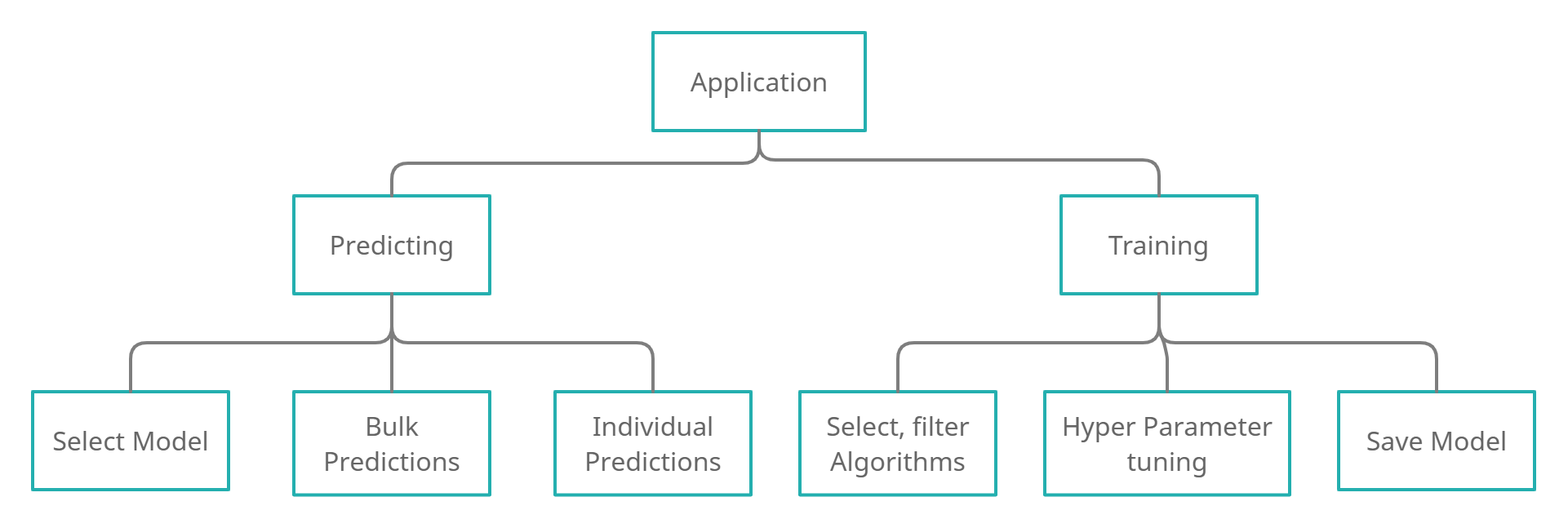


Fig 9: Application overview

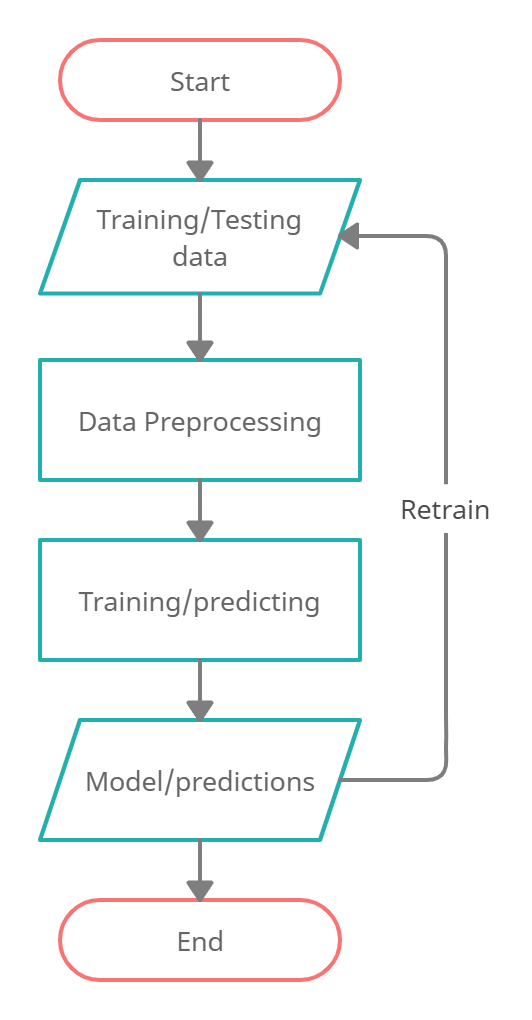


Fig 10: Algorithm Flowchart

## Future work (development)

* Re-training the existing (trained before) models by adding new classes..
* Automating Training with all algorithms and generating detailed report.
* Implementation of Transfer Learning.
* Automated data analysis report for better understanding of data.

horizontal line